Mini-bucket Elimination with Moment Matching

Natalia Flerova, Alexander Ihler, Rina Dechter and Lars Otten
University of California Irvine
[nflerova, ihler, dechter, lotten]@ics.uci.edu

Abstract

We investigate a hybrid of two styles of algorithms for deriving bounds for optimization tasks over graphical models: non-iterative message-passing schemes exploiting variable duplication to reduce cluster sizes (e.g. MBE) and iterative methods that re-parameterize the problem's functions aiming to produce good bounds even if functions are processed independently (e.g. MPLP). In this work we combine both ideas, augmenting MBE with re-parameterization, which we call MBE with Moment Matching (MBE-MM). The results of preliminary empirical evaluations show the clear promise of the hybrid scheme over its individual components (e.g., pure MBE and pure MPLP). Most significantly, we demonstrate the potential of the new bounds in improving the power of mechanically generated heuristics for branch and bound search.

1 Introduction

Graphical models are a popular framework that generalize many combinatorial optimization tasks. In this paper we consider probabilistic graphical models (e.g., Bayesian and Markov networks) [1]. The task of finding the variable assignment maximizing the joint probability of the model is known as the MAP (maximum aposteriori) or MPE (most probable explanation) problem, and is NP-hard.

Mini-Bucket Elimination (MBE) [2] is a popular bounding scheme, which provides an approximation by applying the exact Bucket Elimination (BE) algorithm [3] to a simplified version of the problem obtained by adding duplicates of some variables. If in the MAP assignment of the relaxed problem the duplicates have the same values then this assignment yields the exact solution to the original problem.

The relaxation view of MBE is closely related to the family of iterative approximation techniques based on linear programming (LP) forms of max-product: the "reweighted" max-product algorithm [4], max-product linear programming (MPLP) [5], soft arc consistency [6, 7], etc. [8, 9]. These algorithms can be thought of as "re-parameterizing" or "cost shifting" the original functions, i.e., jointly modifying them in such a way that the original distribution remains unchanged.

In this work we use these ideas to define a new scheme we call mini-buckets with moment matching (MBE-MM). While we do not provide any theoretical guarantees of superiority, our empirical comparison of MBE-MM with pure MBE on various benchmarks shows that MBE-MM achieves better accuracy than pure MBE, while its time overhead is insignificant in most problems. We also compare and contrast MBE-MM with the Max Product Linear Programming (MPLP) algorithm [5]. Our experiments demonstrate that for many benchmarks MPLP can be slow to converge and is less practical than MBE-MM, which acquires its strength from relying on large clusters and can obtain reasonably accurate bounds in one iteration.

Finally, one of the primary uses of MBE is in generating heuristics for best-first and branch and bound search [10, 11]. These have been shown to be quite powerful, and were highly competitive in recent competitions [12, 13]. We show here that the improved scheme of MBE-MM can generate much more powerful heuristics and thus increase the power of Branch and Bound search significantly.

2 Preliminaries and Background on mini-bucket elimination

Consider a set of probability functions **F** over variables **X** defining a graph $G = (X, E)$. Vertices are the variables and an edge connects any two variables appearing in the scope of the same function. The **MAP** task is to find the assignment maximizing the joint probability: $x^* = argmax_{\mathbf{X}} \prod_i f_i$.

A popular algorithm for solving MAP task is **Bucket elimination** (BE) that places each function in the *bucket* of its latest variable according to a certain ordering $o = (X_1, \ldots, X_n)$. For each $Bucket_{X_i}$ noted $\mathbf{B}_i$, from $\mathbf{B}_n$ to $\mathbf{B}_1$, we compute a message $\lambda_i = \max_{X_i} \prod_{j=1}^{n} \lambda_j$, where λ_j are the functions in the $\mathbf{B}_i$, including earlier computed messages. λ_i is placed in the bucket of its latest variable in o. The optimal assignment is recovered in the second, bottom-up phase, when a value is assigned to each variable in o, consulting the functions created during the top-down phase. The time and space complexity of BE are exponential in the graph parameter induced width w.

Mini-bucket elimination (MBE) is an approximation scheme designed to avoid the space and time complexity of BE. Consider a bucket $\mathbf{B}_i$ and an integer bounding parameter z. MBE creates a z-partition $Q = \{Q_1, \ldots, Q_p\}$ of $\mathbf{B}_i$, where each set of functions $Q_j \in Q$, called *mini-bucket*, includes no more than z variables. Then each mini-bucket is processed separately, just as in BE, generating an upper bound on the exact optimizing solution. The time and space complexity of MBE is exponential in z, which is typically chosen to be less than w. In general, greater values of z increase the quality of the bound, untill when $z = w$, MBE finds the exact solution.

3 Background on moment matching strategies

While MBE is usually justified as a relaxation of variable elimination, most iterative re-parameterization approaches are described in terms of solving an LP relaxation of the original model. Wainwright et al. [14, 4] established the connections between LP relaxations of integer programming problems and (approximate) dynamic programming methods using message-passing in the max-product algebra; subsequent improvements in algorithms such as MPLP include coordinate-descent updates that ensure convergence [5, 9].

Without loss of generality MPLP assumes as input the MAP problem for functions θ_{ij} defined over pairs of variables, where the $\theta_{ij} = \log f_{ij}$ are a log-transform of the original functions **F**. The objective is simply the sum of the local functions' maxima, and upper bounds the true optimum:

$$\max_X \sum_{ij} \theta_{ij}(x_i, x_j) \leq \sum_{ij} \max_{x_i, x_j} \theta_{ij}(x_i, x_j), \tag{1}$$

and messages λ_{ij} are used to reparameterize the local functions θ. For space reasons we do not show the derivation of the MPLP, only provide the resulting algorithm in Figure 1. The algorithm iterates updating all edges until convergence, it is guaranteed to improve the objective with each iteration.

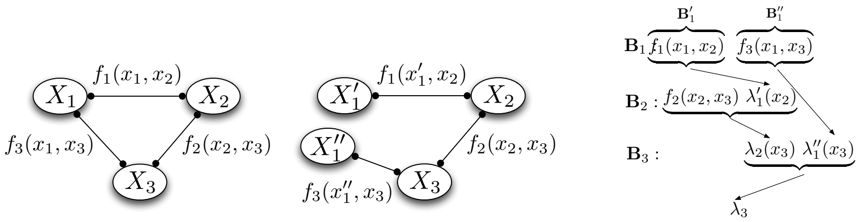
Algorithm 1 Algorithm MPLP

Input: graphical model $\langle \mathbf{X}, \mathbf{D}, \Theta, \sum \rangle$, where θ_{ij} is a potential for each edge $ij \in E$
Output: optimizing assignment x^*

Initialize: $\forall ij, ji \in E$ set $\lambda_{ij}(x_j) = \frac{1}{2} \max_{x_i} \theta_{ij}(x_i, x_j)$
Iterate until convergence:
for all edges $ij, ji \in E$ **do**
  Update:
  $\lambda_{ji}(x_i) = -\frac{1}{2}\lambda_i^{-j}(x_i) + \frac{1}{2} \max_{x_j} [\lambda_i^{-j}(x_i) + \theta_{ij}(x_i, x_j)]$
  where $\lambda_i^{-j}(x_i) = \sum_{k \neq j} \lambda_{ki}(x_i)$
end for
Calculate node beliefs: $b_i(x_i) = \sum_k \lambda_{ki}(x_i)$
Return: the optimizing assignment $x_i^* = \arg\max_{x_i} b(x_i)$

4 Mini-bucket elimination with moment-matching

The source of inaccuracy in MBE is the independent processing of mini-buckets $\{Q_1, \ldots, Q_p\}$ of $\mathbf{B}_i$, which is equivalent to creating duplicates of variable X_i: $\{X_{i_1}, \ldots, X_{i_p}\}$ and then exactly processing the new buckets $\{\mathbf{B}_{i_1}, \ldots, \mathbf{B}_{i_p}\}$. Given the assignment $\mathbf{X}^*$ found by MBE algorithm, we



(a) Original problem graph (b) Problem graph with duplicated variables (c) The flow of messages in the bucket

Figure 1: Example

can show that if all optimizing values of the duplicates of each variable take the same value, then the solution found by MBE is exact.

One idea for increasing the accuracy of the solution is to use *moment-matching*, an idea closely related to the notions of cost-shifting ([6], [15]). We use the simple example in Figure 1a to illustrate the idea underlying moment matching. Applying MBE-MM with $z = 2$ along ordering $o = \{X_3, X_2, X_1\}$ to the problem in Figure 1a results in partitioning of the functions into mini-buckets as shown in Figure 1c. As in cost-shifting for soft arc-consistency [6] the function of bucket $\mathbf{B}_1$ can be devided and muptiplied some non-negative function $g(x_1)$ without changing the expression, yielding:

$$C_1(x_2, x_3) = \underset{x_1}{max} f_1(x_1, x_2) \cdot f_3(x_1, x_3) = \underset{x_1}{max} f_1(x_1, x_2) g(x_1) \cdot f_3(x_1, x_3) / g(x_1) \quad (2)$$

Bucket $\mathbf{B}_1$ is split into two mini-buckets: $\mathbf{B}_1'$ and $\mathbf{B}_2''$, resulting in the problem graph of Figure 1b, and we can write the upper bound on the function in $\mathbf{B}_i$ $\hat{C}$ as:

$$C_1(x_2, x_3) \leq \hat{C}_1(x_2, x_3) = \max_{x_1'} f_1(x_1', x_2) g(x_1') \cdot \max_{x_1''} f_3(x_1'', x_3) / g(x_1'') \quad (3)$$

We would like for the optimal values of the two buckets to agree, $x_1^* = x_1^{*\prime} = x_1^{*\prime\prime}$, since if this condition holds for the full MBE solution, it will be optimal. Although we have not yet seen the rest of the functions (e.g., $f_2(x_2, x_3)$), if we assume these functions are uninformative we can enforce agreement by selecting

$$g(x_1) = \sqrt{\max_{x_3} f_3(x_1, x_3) \;/\; \max_{x_2} f_1(x_1, x_2)}. \quad (4)$$

The functions $\max_{x_2} f_1(x_1, x_2)$ and $\max_{x_3} f_3(x_1, x_3)$ are called *max-marginals* of the functions f_1, f_3, and (4) ensures that these max-marginals agree in the new parameterization, $f_1 \cdot g$ and f_3/g.

The MBE relaxation in Figure 1b can be shown to be equivalent to a Lagrangian relaxation of the original problem, and thus to the set of LP relaxations optimized by many variants of max-product [8]. Moreover, when taken on the original graph our moment-matching updates are equivalent to a particular schedule of fixed point updates in the LP dual formulation of the MAP problem. These updates can be viewed as coordinate descent on the upper bound given by independent maximization, (1); see for example [16]. However, since the influence of later functions is not yet known when the matching step is performed, the single-pass algorithm MBE-MM is not necessarily guaranteed to improve on the original MBE bound.

The major difference between MBE and LP relaxations is thus primarily in the decisions of what variable scopes will be used, and in the amount of iterative tightening performed. At one end, MBE is non-iterative but typically uses functions over many variables, whose scopes are easily selected using heuristics and the z-bound parameter. In contrast, LP relaxations typically work on the original graph, performing many iterations to tighten the bound; extensions to these methods may tighten the bound by incrementally increasing the function sizes slightly, using heuristics to determine which scopes to include [16]. MBE-MM is thus a single-pass bound that uses the iterative viewpoint to inform its heuristic decisions. The algorithm is presented in Algorithm 2.

Algorithm 2 Algorithm MBE-MM

Input: An optimization task $\mathcal{P} = (\mathbf{X}, \mathbf{D}, \mathbf{F}, \prod, max)$; An ordering of variables $o = \{X_1, \ldots, X_n\}$; parameter z.
Output: bounds on the MAP cost and the corresponding assignment for the expanded set of variables (i.e., node duplication).

Initialize: Generate an ordered partition of functions $\mathbf{F} = \{f_1, \ldots, f_j\}$ into buckets $\mathbf{B}_1, \ldots, \mathbf{B}_n$, where $\mathbf{B}_i$ along o.
Backward:
for $i \leftarrow n$ down to 1 (Processing bucket $\mathbf{B}_i$) **do**
  Partition functions in bucket $\mathbf{B}_i$ into $\{Q_{i_1}, \ldots, Q_{i_p}\}$, where each Q_{i_k} has no more than z variables.
  Find the set of variables common to all the mini-buckets: $S_i = S_{i_1} \cap \cdots \cap S_{i_p}$, where $S_{i_k} = var(Q_{i_k})$
  Find the function of each mini-bucket Q_{i_k}: $F_{i_k} \leftarrow \prod_{f \in Q_{i_k}} f$
  Find the max-marginals of each mini-bucket Q_{i_k}: $\mu_{i_k} = max_{var(Q_{i_k})/S_i}(F_{i_k})$
  Update functions of each mini-bucket Q_{i_k}: $F_{i_k} \leftarrow F_{i_k} \cdot \sqrt[k]{\mu_{i_1} \cdot \cdots \cdot \mu_{i_p}}/\mu_{i_k}$
  Generate messages $\lambda_{i_k} = max_{X_i} F_{i_k}$ and place each in the largest index variable in $var(Q_{i_k})$
end for
Return: The set of all buckets, and the vector of m-best costs bounds in the first bucket.

5 Empirical results

In our empirical evaluation we investigate the impact of moment-matching and other cost-shifting schemes (e.g. h-MBE [15]) on the mini-bucket algorithm. We also compare MBE-MM with MPLP.

We experimented with two sets of instances, containing selected pedigree (Figure 2) and Weighted CSP instances (Figure 4) from the UAI 2008 evaluation [12]. We solve the MAP task for all the instances. One factor that can influence the performance of MBE significantly is the way functions are partitioned into the mini-buckets. The issue was extensively studied by Rollon and Dechter [17], who introduced and evaluated a set of partitioning heuristics that we use in our experiments.

5.1 Impact of moment-matching on the accuracy of the bound

Figure 2 presents the upper bounds computed by MBE with and without moment-matching (denoted MBE-MM and MBE) and by h-MBE on the pedigrees with z-bound=10. The first two schemes use two partitioning heuristics: scope-based and content-based with l2 distance measure (see [15] for details). The h-MBE uses scope-based partitioning. We see that both cost-shifting methods, h-MBE and MBE-MM produce better bounds than the pure MBE with no moment-matching. The figure also shows the corresponding runtimes (sec) of the MBE-MM and MBE. The runtime of the h-MBE scheme is omitted due to drastic differences in implementation that renders speed comparison meaningless.

5.2 The impact of iterations (MPLP)

As can be seen from [8] and [18], MBE-MM applied to original factors is equivalent to a single iteration of MPLP. Algorithm MPLP improves on this approach by running multiple updates, decreasing the bound with each iteration. The MBE-MM scheme, on the other hand, can influence accuracy by combining factors into larger clusters. Both of these enhancement schemes increase their runtime. In our experiments we explore which method trades time for accuracy more effectively.

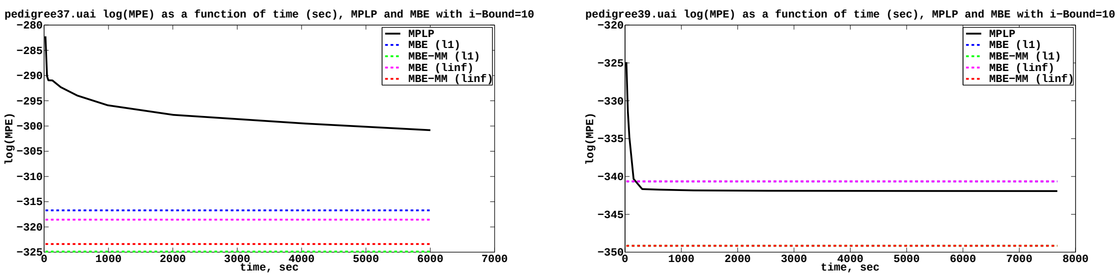
Figure 3 illustrates the typical behavior of algorithms on selected pedigrees, presenting the dependence of the upper bound on the log(MPE) on time for MPLP, compared against MBE-MM and pure MBE. Since MBE algorithms are not iterative, the results do not change with the time. Note that in these figures we plot the results for MBE-MM and MBE with z-bound=10. The cutoff for the MPLP algorithms was 1500 iterations.

We can see that even though MPLP algorithm improves the bound with more time, as theory suggests, it can not achieve the same accuracy as MBE-MM with given z-bounds. It shows that the orthogonal use of large cluster can yield far better accuracies even though MBE-MM is not iterative.

In Table 4 we see the upper bounds produced by MBE-MM with two content-based heuristics using l2 and linf distance measures and z-bound=10 and MPLP ran for 5, 500 and 1500 iterations for select WCSP instances. We see that even for a large number of iterations MPLP does not achieve the same accuracy as MBE-MM for more than half of these instances.

Instances	n	k	w	MBE scope heuristic		MBE l2 heuristic		h-MBE
				with MM log(MPE) time(sec)	**no MM** log(MPE) time(sec)	**with MM** log(MPE) time(sec)	**no MM** log(MPE) time(sec)	log(MPE)
pedigree1.uai	298	4	15	-104.3317 1.043	-103.8327 0.827	-104.3453 1.51	-104.0717 1.019	-104.801258
pedigree7.uai	867	4	28	-251.2962 3.13	-247.1016 2.273	-251.6741 3.973	-252.9009 3.396	-250.083186
pedigree9.uai	935	7	25	-269.8636 3.194	-263.5919 2.369	-269.1398 4.496	-264.0459 3.505	-269.68553
pedigree13.uai	888	3	30	-158.3137 3.039	-156.9928 2.139	-158.4953 4.05	-157.7176 3.283	-104.801258
pedigree19.uai	693	5	21	-203.5111 3.72	-197.8175 2.367	-202.6335 5.602	-199.2458 3.808	-200.366564
pedigree20.uai	387	4	20	-118.3357 1.403	-114.7234 0.957	-117.7419 2.019	-116.0857 1.652	-116.049293
pedigree23.uai	309	5	21	-140.7592 1.583	-138.9552 0.948	-141.6805 2.803	-139.0275 1.562	-142.253279
pedigree31.uai	1006	5	29	-290.5953 3.733	-283.8814 3.075	-289.9581 4.755	-286.074 3.611	-289.030586
pedigree37.uai	726	5	20	-323.97 3.007	-316.8191 2.114	-324.8728 5.227	-318.3267 4.281	-320.589194
pedigree38.uai	581	5	16	-193.9027 5.59	-190.9309 2.549	-196.6335 13.594	-195.4372 7.512	-196.125622
pedigree39.uai	953	5	20	-348.4941 4.003	-340.3035 2.681	-349.1526 5.458	-340.6459 4.219	-343.058959
pedigree41.uai	885	5	29	-265.3413 4.136	-253.4084 2.916	-265.9065 5.607	-265.1594 4.13	-265.642738
pedigree50.uai	478	6	16	-141.221 19.945	-139.7527 6.607	-141.4497 28.057	-140.0774 20.161	-141.511359
pedigree51.uai	871	5	33	-236.7164 3.682	-222.4415 2.579	-235.9544 5.568	-229.6678 3.969	-236.635487

Figure 2: Upper bounds and runtime (sec) for pedigree instances computed by MBE with and without moment matching (MM), using scope-based and l2-distance partitioning heuristics with z-bound=10. For each instance we report number of variables n, largest domain size k and the induced width along the ordering used w. We also report the bound found by h-MBE. The runtimes of the h-MBE are not included due to the difference in implementation.



(a) (b)

Figure 3: Upper bounds on log(MPE) as a function of time for selected pedigrees. We plot MPLP ran on the original factors, MBE and MBE-MM with z-Bound=10 and partitioning heuristics with distance measures l1 and linf. MBE and MBE-MM are not iterative, so their result doesn't change with time. MPLP ran for 1500 iterations. NB: for pedigree39 the results for l1 and linf overlap.

5.3 MBE-MM and MPLP as search guiding heuristic

One of the most popular applications of bounding schemes is generating heuristics for informed search algorithms. We tested pure MBE, MBE-MM and MPLP algorithms as heuristic generators for the well-known AND/OR Branch and Bound algorithm [19] on pedigree, grid, WCSP and mastermind instances for various z-bounds. We used scope-based partitioning for both MBE and MBE-MM. Figure 7 shows the anytime results for the AOBB with different heuristic generators. For each time cutoff we report the number of instances for which algorithm obtained any solution, the number of instances, for which an exact solution was found, but not yet proved to be optimal, and a number of instances for which the optimality of solution was proved. For each time interval we show the results in bold only when all 3 numbers are higher than for the competing schemes. Figures 5 and 6 show the runtimes in second and the number of nodes expanded by the AOBB

Instance	n	k	w	MBE-MM		MPLP		
				l2	linf	5 iter	500 iter	1500 iter
1502.uai	209	4	6	**-2.8954**	**-2.8954**	-2.6753	-2.6886	-2.6886
29.uai	82	4	14	**-3.6906**	-3.6888	-3.2006	-3.2259	-3.2259
404.uai	100	4	19	**-5.2229**	-5.0545	-3.5222	-3.7092	-3.7432
408.uai	200	4	35	-3.1147	-3.1177	-3.6974	-3.9934	**-4.0735**
42.uai	190	4	26	**-3.1872**	-3.0472	-2.1092	-2.3906	-2.5227
503.uai	143	4	9	-3.1872	-3.1872	-2.9683	-3.2905	**-3.4497**
505.uai	240	4	22	-1.1207	-2.1888	-2.7076	-3.0725	**-3.2433**
54.uai	67	4	11	**-3.0701**	-2.9848	-1.8466	-2.0719	-2.0812

Figure 4: The upper bounds on the log(MPE) for the select WCSP instances by MBE-MM with two content-based heuristics using l2 and linf distance measures with z-bound=10 and MPLP ran for 5, 500 and 1500 iterations. For each instance we report the number of variables n, the largest domain size k and the induced width along the ordering used w. The best bounds are shown in bold.

with MBE, MBE-MM and MPLP heuristic generators for selected pedigree and grids instances for various z-bound.

We can see that neither of the bounding schemes generates heuristic information that would allow the search to produce consistently better results. The search algorithm that uses MBE-MM in most cases produce better results than the one that uses pure MBE. Notable exception is the set of WCSP instances, where algorithm with MBE heuristic consistently performs the best. MPLP and MBE-MM take turns in producing better results, depending on the time cut-off, cluster sizes and instance set.

Instances (n,k,w,h)	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=4	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=6	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=8	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=10
pedigree7 867,4,32,90	— — —	— 46261 / 6414458757 54000 / 8115757656	— 5993 / 929988636 13211 / 1967808000	— 1987 / 303782644 4975 / 728673440
pedigree13 888,3,32,102	— — —	— — —	— — —	— 57583 / 8816940735 70328 / 10533681283
pedigree20 387,5,22,60	4460 / 838691448 10805 / 1882064728 11203 / 1880718191	167 / 35218516 378 / 73141782 582 / 105186811	137 / 30109086 112 / 24006114 262 / 48856344	44 / 6894997 25 / 5353765 87 / 15256293
pedigree9 935,7,27,100	— — —	— — —	— 7086 / 1209510942 9397 / 1366185532	46434 / 7509543280 1206 / 207241642 2161 / 315235125
pedigree50 478,6,17,47	25440 / 4483574892 — —	39 / 7903081 47 / 9373215 12146 / 1862620741	16 / 3817267 11 / 2066749 886 / 127499702	6 / 616641 17 / 99160 46 / 4451969
pedigree23 309,5,25,51	45 / 8621377 31 / 5539840 89 / 15065580	22 / 4525816 3 / 737573 20 / 3472051	13 / 2204965 2 / 301362 24 / 4133711	4 / 669332 0 / 46830 9 / 1606108
pedigree37 726,5,21,56	298 / 48846178 174 / 19065653 145 / 16043993	33 / 7774713 8 / 2012510 6 / 1416885	13 / 2996251 0 / 214882 1 / 166267	6 / 1594382 1 / 36185 0 / 30353
pedigree33 581,4,28,98	24142 / 3699778889 1958 / 287752998 2076 / 312108999	201 / 28902650 260 / 36529410 287 / 40331678	177 / 29051952 7 / 1125722 8 / 1405533	89 / 20884397 5 / 992393 8 / 1448901
pedigree30 1015,5,21,108	13198 / 2320333183 — —	1690 / 298201914 442 / 65590061 508 / 71466803	246 / 43567736 36 / 5423758 99 / 14000719	109 / 19246868 21 / 3007912 34 / 4760013
pedigree39 953,5,21,76	30724 / 4849893762 8315 / 1262698139 9100 / 1271875710	2871 / 493752913 294 / 51962281 377 / 61178785	732 / 127985658 29 / 5224288 26 / 4188690	136 / 23482266 15 / 2745544 17 / 2776079
pedigree25 993,5,25,69	— 13321 / 1779583093 4670 / 597669088	1303 / 212051451 36 / 5190620 48 / 6456274	145 / 25395001 4 / 559510 3 / 474334	58 / 9218653 0 / 66674 1 / 161821
pedigree1 298,4,15,48	0 / 74793 0 / 135762 42 / 6435018	0 / 143562 0 / 14223 11 / 1988109	1 / 64978 0 / 3833 7 / 1142934	0 / 3754 1 / 1901 4 / 744348

Figure 5: Runtime (sec) / number of nodes expanded for pedigree instances by AOBB with MBE, MBE-MM or MPLP as a heuristic generator. For each instance we report number of variables n, largest domain size k and the induced width along the ordering used w.

Instances (n,k,w,h)	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=3	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=5	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=10	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=15
50-16-5 256,2,21,79	14047 / 3174893721 11918 / 2442554277 7243 / 1600369401	6759 / 1495170158 257 / 62304128 209 / 51010597	97 / 28789854 1 / 173174 1 / 295211	3 / 1188559 0 / 20391 1 / 86023
50-20-5 400,2,27,97	— — —	— 28985 / 6530721824 6529 / 1477984347	3589 / 953505413 11 / 3125798 26 / 6981464	385 / 106703929 1 / 178389 6 / 1547636
75-16-5 256,2,21,73	2457 / 566137206 511 / 111054314 516 / 104399258	245 / 55464972 32 / 8455201 37 / 9886997	7 / 2065927 0 / 56708 1 / 86972	0 / 190759 0 / 8749 0 / 13694
75-20-5 400,2,27,99	— — 47557 / 10078495991	— 25258 / 4544121013 8458 / 1518974618	1912 / 422432794 6 / 1693491 13 / 3330988	7 / 1686365 1 / 11539 1 / 14719
90-20-5 400,2,27,99	7281 / 1611969572 5575 / 1206417015 3162 / 685514805	1199 / 291338041 585 / 136441594 389 / 83159717	14 / 3765984 1 / 103457 0 / 122851	0 / 85134 0 / 2461 0 / 4520
90-21-5 441,2,28,106	7722 / 1561888432 9064 / 1885446239 4709 / 919156950	1585 / 323191066 861 / 182302009 593 / 128038384	15 / 4004517 0 / 111099 1 / 146184	1 / 373433 0 / 6381 0 / 7314
90-22-5 484,2,30,109	27283 / 4804835455 17130 / 3159832586 10279 / 1903308709	2327 / 454068262 1172 / 219854528 604 / 112967966	50 / 11323811 6 / 1402416 1 / 187789	3 / 823393 0 / 9998 1 / 8715
90-26-5 676,2,36,136	— 70798 / 11832076161 58797 / 10041886801	36469 / 6252167622 7077 / 1267186678 4000 / 724913557	386 / 79457203 21 / 4464564 16 / 3445275	52 / 12386883 2 / 231824 2 / 240945

Figure 6: Runtime (sec) / number of nodes expanded for grid instances by AOBB with MBE, MBE-MM or MPLP as a heuristic generator. For each instance we report number of variables n, largest domain size k and the induced width along the ordering used w.

Instances	z-bound	Heuristic	1 sec	5 sec	10 sec	1 min	5 min	1 h	24 h
pedigrees	10	MBE	17:2:1	20:7:2	21:8:4	21:11:8	21:12:10	22:14:11	22:17:14
		MBE-MM	**21:10:4**	**21:12:6**	**22:13:6**	**22:13:11**	**22:15:12**	22:15:14	**22:19:18**
		MPLP	17:7:2	20:7:4	20:9:6	22:13:9	22:14:10	22:16:12	22:18:18
	8	MBE	15:2:1	16:3:1	16:5:1	19:7:4	20:9:9	20:13:11	22:16:14
		MBE-MM	18:4:2	**20:7:4**	**20:7:5**	21:12:9	21:13:11	21:14:12	21:18:16
		MPLP	15:5:1	17:6:2	19:6:4	22:10:7	22:11:9	22:16:10	**22:19:17**
	6	MBE	10:2:1	14:3:1	16:3:1	17:6:4	17:8:6	20:11:11	21:14:11
		MBE-MM	**14:3:1**	17:3:2	**19:4:3**	21:6:6	**21:10:8**	21:13:11	21:17:13
		MPLP	13:3:0	16:4:0	19:4:1	20:7:5	21:9:6	22:11:9	22:16:13
	4	MBE	7:1:0	10:1:0	11:1:0	14:1:1	15:2:1	18:6:4	19:8:7
		MBE-MM	**9:2:1**	**11:2:1**	11:2:1	14:2:2	16:4:4	18:7:5	19:10:9
		MPLP	9:1:0	10:1:0	12:1:0	17:3:1	**20:4:4**	**21:8:5**	21:11:8
grids	15	MBE	21:9:9	23:17:12	24:17:14	26:21:18	27:21:20	27:22:24	27:24:27
		MBE-MM	**27:22:21**	**28:22:24**	**28:23:24**	**29:24:25**	**29:25:27**	**29:25:28**	**29:26:29**
		MPLP	26:20:16	28:21:22	28:21:24	28:22:25	28:22:25	28:23:27	29:25:29
	10	MBE	15:3:3	18:7:3	19:8:7	21:12:10	21:15:12	24:20:18	24:22:24
		MBE-MM	**22:12:11**	23:17:11	**23:19:17**	**24:20:20**	**25:21:23**	**26:23:25**	**27:24:27**
		MPLP	**22:12:11**	23:16:12	23:17:14	24:20:19	**25:21:23**	**26:23:25**	**27:24:27**
	5	MBE	8:1:1	9:4:1	9:4:1	9:6:3	10:7:5	14:12:11	15:15:15
		MBE-MM	**10:2:1**	**11:3:1**	**12:3:2**	14:7:4	18:8:7	18:14:11	19:19:19
		MPLP	9:2:1	**11:3:1**	11:3:2	**15:7:5**	**18:9:8**	**19:14:11**	**20:20:20**
	3	MBE	3:0:0	7:1:0	7:1:1	8:1:1	9:3:2	10:8:5	11:11:11
		MBE-MM	7:1:1	8:1:1	8:1:1	9:3:3	9:5:4	12:8:6	13:13:13
		MPLP	**7:3:1**	**9:3:1**	**9:3:1**	**10:5:3**	**11:8:4**	**14:10:8**	**15:15:15**
mastermind	15	MBE	**128:128:56**	**128:128:61**	**128:128:72**	**128:128:88**	**128:128:107**	**128:128:128**	**128:128:128**
		MBE-MM	128:128:33	128:128:38	128:128:42	128:128:73	128:128:96	128:128:120	**128:128:128**
		MPLP	96:96:1	116:116:16	124:124:18	125:125:35	126:126:56	126:126:94	126:126:126
	10	MBE	126:126:19	128:128:25	128:128:37	**128:128:61**	**128:128:90**	128:128:113	**128:128:128**
		MBE-MM	**128:128:27**	**128:128:34**	**128:128:42**	128:128:60	128:128:88	**128:128:119**	**128:128:128**
		MPLP	92:92:0	103:103:16	116:116:17	128:128:33	128:128:59	128:128:93	128:128:112
	5	MBE	**102:102:46**	**105:105:54**	**105:105:60**	**105:105:75**	**105:105:88**	**105:105:98**	**105:105:105**
		MBE-MM	92:92:19	103:103:22	105:105:25	105:105:45	105:105:67	105:105:79	**105:105:105**
		MPLP	58:58:0	73:73:9	82:82:16	97:97:26	105:105:44	105:105:56	105:105:81
	3	MBE	**94:94:27**	**100:100:37**	**105:105:51**	**105:105:68**	**105:105:75**	**105:105:93**	**105:105:104**
		MBE-MM	65:65:0	75:75:15	88:88:16	104:104:17	105:105:22	105:105:61	105:105:79
		MPLP	53:53:0	59:59:1	68:68:13	78:78:16	90:90:17	105:105:38	105:105:73
WCSP	5	MBE	**6:6:4**	**6:6:4**	**6:6:5**	**6:6:6**	**6:6:6**	**6:6:6**	**6:6:6**
		MBE-MM	5:5:4	5:5:4	5:5:5	5:5:5	5:5:5	5:5:5	5:5:5
		MPLP	5:5:4	5:5:4	5:5:5	5:5:5	5:5:5	5:5:5	5:5:5

Figure 7: Anytime results for the AOBB with different heuristic for pedigree, grid, mastermind and WCSP instances for various z-bound. For each time cutoff we report the 3 numbers: the number of instances for which algorithm obtained any solution, the exact solution or for which the optimality of solution was proved. For example, the expression ”20:7:2” in the first row of the 3rd column means that in 5 seconds, MBE with z-bound=10 found any solutions (possibly suboptimal) for 20 instances, found exact solution for 7 out of them and proved the optimality of the solution for 2.

6 Conclusion

We presented Mini-bucket elimination with moment-matching, a new bounding scheme for optimization tasks in graphical model. We discussed the connection between moment-matching in MBE-MM and methods used in the previously developed algorithms: a) shifting costs procedure, used, for example, in horizontal MBE [15], Max-sum diffusion [20] or Soft arc-consistency algorithm [6]; b) update in the MPLP, which is derived as a step in the block coordinate descent in the dual of the LP relaxation of the original problem. We demonstrated empirically that moment-matching improves MBE performance across all instances and for any partitioning heuristic (we only showed two schemes here for lack of space, but our results were consistently better). We also demonstrated that in many cases MBE-MM can find a more accurate bound than MPLP faster, even for small z-bounds, and has a performance comparable with horizontal-MBE presented earlier by [15]. The most impressive aspect is the ability to improve search algorithm with heuristic function that do not require more computational power (i.e., when z is fixed). Future work includes developing of a hybrid scheme that would use the output of MBE-MM as a starting point for the MPLP algorithm this extending MPLP to be executed over the mini-bucket clusters.

References

[1] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

[2] R. Dechter and I. Rish. Mini-buckets: A general scheme for bounded inference. *Journal of the ACM (JACM)*, 50(2):107–153, 2003.

[3] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1):41–85, 1999.

[4] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, 51(11):3697–3717, 2005.

[5] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. *Advances in Neural Information Processing Systems*, 21(1.6), 2007.

[6] T. Schiex. Arc consistency for soft constraints. *Principles and Practice of Constraint Programming (CP2000)*, pages 411–424, 2000.

[7] S. Bistarelli, R. Gennari, and F. Rossi. Constraint propagation for soft constraints: Generalization and termination conditions. *Principles and Practice of Constraint Programming–CP 2000*, pages 83–97, 2000.

[8] J.K. Johnson, D.M. Malioutov, and A.S. Willsky. Lagrangian relaxation for map estimation in graphical models. *Arxiv preprint arXiv:0710.0013*, 2007.

[9] David Sontag and Tommi Jaakkola. Tree block coordinate descent for MAP in graphical models. In *AI & Statistics*, pages 544–551. JMLR: W&CP 5, 2009.

[10] Radu Marinescu and Rina Dechter. AND/OR Branch-and-Bound search for combinatorial optimization in graphical models. *Artif. Intell.*, 173(16-17):1457–1491, 2009.

[11] Radu Marinescu and Rina Dechter. Memory intensive AND/OR search for combinatorial optimization in graphical models. *Artif. Intell.*, 173(16-17):1492–1524, 2009.

[12] A. Darwiche, R. Dechter, A. Choi, V. Gogate, and L. Otten. Results from the probablistic inference evaluation of UAI08, a web-report in http://graphmod.ics.uci.edu/uai08/Evaluation/Report. *In: UAI applications workshop*, 2008.

[13] Gal Elidan and Amir Globerson. UAI 2010 approximate inference challenge. `http://www.cs.huji.ac.il/project/UAI10/`.

[14] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudomoment matching. In *Workshop on Artificial Intelligence and Statistics*, volume 21. Citeseer, 2003.

[15] E. Rollon and J. Larrosa. Mini-bucket elimination with bucket propagation. *Principles and Practice of Constraint Programming-CP 2006*, pages 484–498, 2006.

[16] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. 2010.

[17] E. Rollon and R. Dechter. New mini-bucket partitioning heuristics for bounding the probability of evidence. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI2010)*, pages 1199–1204, 2010.

[18] Q. Liu and A. Ihler. Bounding the partition function using hölders inequality. 2011.

[19] R. Marinescu and R. Dechter. And/or branch-and-bound for graphical models. In *International Joint Conference on Artificial Intelligence*, volume 19.

[20] V.A. Kovalevsky and V.K. Koval. A diffusion algorithm for decreasing energy of max-sum labeling problem. *Unpublished, approx*, 1975.